REMARKS/ARGUMENTS

Claims 1-28 are pending in the present application.

This Amendment is in response to the final Office Action mailed February 1, 2011 to support a Request for Continued Examination (RCE) filed concurrently. In the final Office Action, the Examiner objected to claims 3-7, rejected claims 8-14 under 35 U.S.C. §101; claims 1-28 under 35 U.S.C. §103(a). Applicant has amended claims 1, 8, 15, and 22. Reconsideration in light of the amendments and remarks made herein is respectfully requested.

Rejection Under 35 U.S.C. § 101

In the Office Action, the Examiner rejected claims 8-14 under 35 U.S.C. §101 because the claimed invention is directed to non-statutory subject matter. Specifically, the Examiner contends that "[s]uch components are only software modules and thus can be interpreted as software program listing per se." (Final Office Action, page 6, paragraph 9). Applicant respectfully disagrees for the following reasons.

First, the program transformer is described in paragraphs [0066] – [0069] in connection with Figure 9. The description specifically states that "[t]he program transformer 135 may be implemented by software, hardware, firmware, or any combination of these elements." Accordingly, the Examiner's statement that "[s]uch components are only software modules" is improper because it is clear that the program transformer 135 may be implemented by software, hardware, firmware, or any combination of these elements. The use of the alternative "or" indicates that the program transformer 135 may be implemented by software, OR by hardware, OR by firmware, or any combination of these elements (emphasis added.) Accordingly, it is not limited to only software implementation.

Second, even if the program transformer 135 is implemented by software, it does not mean that it can be interpreted as software program listing per se as alleged by the Examiner. As described in the specification, the program transformer 135 may be executed by the processor to compile, translate, or transform the program code. See, for example, Specification, paragraph [0031], [0066] – [0069], and Figure 9. Accordingly, it is not a program *listing* per se. A program listing is exactly what it is called, a listing of a program. In other words, it is a source code of the program. A listing is merely a piece of paper, or embodied in non-functional

medium. In contrast, the program transformer as described in the specification and as claimed is capable of performing operations. For example, claim 8 recites "associate blocks of instructions . . . " and "sink the wait instruction down the block globally . . ." Certainly, a mere listing of a program cannot associate blocks of instructions or sink the wait instruction. It is therefore not a non-functional program listing.

In the Final Office Action, the Examiner contends that the claim language merely recites 'associate blocks and 'sink the wait instruction,' but does not explicitly define the blocks or instructions are in the computer memory or perform operation based on the blocks/instructions in the memory (Final Office Action, page 3, lines 3-6). Applicant respectfully disagrees for the following reasons.

Claims should be interpreted consistently with the specification, which provides content for the proper construction of the claims because it explains the nature of the patentee's invention. See *Renishaw PLC v. Marposs Societa' per Azioni*, 158 F.3d 1243,1250, 48 USPQ 2d (BNA) 1117 (Fed. Cir. 1998). MPEP 2111. Here, the specification provides ample support for the claim language. Paragraph [0031] explicitly states that the main memory 130 includes a compiler to perform latency hiding using a coloring approach (Specification, paragraph [0031], lines 7-8; Fig. 1A, element 135). In addition, the specification describes the program transformer 135 in details. See, for example, paragraphs [0066] – [0069] and Fig. 9.

Accordingly, Applicant respectfully requests the rejections under 35 U.S.C. §101 be withdrawn.

Rejection Under 35 U.S.C. § 103

In the final Office Action, the Examiner rejected claims 1-28 under 35 U.S.C. §103(a) as being unpatentable over Chang (Chang et al., IMPACT: An architectural Framework for Multiple-Instruction-Issue Processors") ("Chang") in view of U.S. Publication No. 2005/0149916 A1 issued to Shpeisman et al. ("Shpeisman") and further in view of Midkiff (Midkiff et al., Compiler Algorithms for Synchronization) ("Midkiff"). Applicant respectfully traverses the rejection and submits that the Examiner has not met the burden of establishing a prima facie case of obviousness.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. *MPEP §2143, p. 2100-126 to 2100-130 (8th Ed., Rev. 5, August 2006).* Applicant respectfully submits that there is no suggestion or motivation to combine their teachings, and thus no *prima facie* case of obviousness has been established.

Furthermore, the Supreme Court in *Graham v. John Deere*, 383 U.S. 1, 148 USPQ 459 (1966), stated: "Under § 103, the scope and content of the prior art are to be determined; differences between the prior art and the claims at issue are to be ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background, the obviousness or nonobviousness of the subject matter is determined." MPEP 2141. In *KSR International Co. vs. Teleflex, Inc.*, 127 S.Ct. 1727 (2007) (Kennedy, J.), the Court explained that "[o]ften, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue." The Court further required that an explicit analysis for this reason must be made. "[R]ejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness." *KSR 127 S.Ct.* at 1741, quoting *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006). In the instant case, Applicant respectfully submits that there are significant differences between the cited references and the claimed invention and there is no apparent reason to combine the known elements in the manner as claimed, and thus no *prima facie* case of obviousness has been established.

1. Claims 1-28:

Chang discloses an architectural framework for multiple-instruction issue processors. Prepass code scheduling is performed prior to register allocation while postpass code scheduling is performed after register allocation (Chang, page 268, section 2.4 "Code Scheduling

Algorithm"). Both prepass and postpass code scheduling algorithms consist of the following steps: 1) form traces from basic blocks that are likely to be executed as a sequence; 2) form a large superblock from each trace of basic blocks by code duplication. A superblock has a unique entry point, and one or more exit points (Chang, page 268, section 2.4 "Code Scheduling Algorithm"). The code scheduler moves code both upward and downward across branch operations within a superblock (Chang, page 268, section 2.5 "Code Scheduling Models", first paragraph). For downward code motion, e.g., X precedes Y, if Y does not depend on X then X can be moved below Y (Chang, page 268, section 2.5 "Code Scheduling Models", second paragraph).

Shpeisman discloses data layout mechanism to reduce hardware resource conflicts. compiler 202 may create a data layout 800 by mapping data elements 352, 354, 356, 358 referenced by the machine-executable instructions 328, 330, 332, 334, 336, 338 to memory locations 214 in accordance with a colored conflict graph 700 (Shpeisman, paragraph [0034], lines 1-7). Each color C1, C2, C3 may correspond to a hardware resource (Shpeisman, paragraph [0058], lines 1-3). The assignment of a color to a node may represent the assignment of a hardware resource to the data represented by the node (Shpeisman, paragraph [0058], lines 5-8).

Midkiff discloses compiler algorithms for synchronization. Two instruction sets are used to synchronize between groups of statement (Midkiff, page 1487, left column, Section IV.C "Two synchronization Instruction Sets"). The set instruction is used to signal that some event has occurred, and the wait instruction is used to wait until that event occurs (Midkiff, page 1487, right column, Section IV.C "Two synchronization Instruction Sets"). When synchronizing dependencies with wait and set, a set statement is placed after the source of the dependence to signal that the source has executed, and a wait is placed before the sink to ensure that the source executes first (Midkiff, page 1488, right column, Section V.B "Generation of wait and set instructions")

Chang, Shpeisman, and Midkiff, taken alone or in any combination, do not disclose or render obvious, at least one of: (1) associating blocks of instructions between start and end of a critical section with color information, the blocks corresponding to a program trace and containing a wait instruction associated with a memory access; and (2) sinking the wait

instruction down the blocks globally across the blocks to the end of the critical section using the color information and a dependence constraint on the wait instruction.

First, Chang merely discloses a large superblock from each trace of basic blocks formed by code duplication (Chang, page 268, section 2.4 "Code Scheduling Algorithm"), not a critical section. A superblock is merely formed by duplicating code from the traces of the basic blocks. In contrast, a critical section is a section that is protected to ensure mutual exclusiveness.

The Examiner acknowledges the information for the critical section (Final Office Action, page 3, lines 14-15). However, the Examiner states that although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims, citing *In re Van Guens*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993) (Final Office Action, page 3, lines 15-18). The Examiner concludes that the critical section as recited in the claim can be reasonably interpreted as code block. Applicant respectfully disagrees for the following reasons. The term "critical section" is well known in the art of operating systems, compiler, and synchronization. It refers to a program section that can only be executed one at a time, or a program code that is mutually exclusive. Accordingly, the plain meaning of a critical section indicates a "mutually exclusive" section. This is not a limitation from the specification that is read into the claims. Furthermore, it appears that the Examiner misread *Van Guens*. In *Van Guens*, the claim in question recites a magnet assembly with a "uniform magnetic field". The board found that the Japanese reference disclosed a magnetic assembly with a substantially uniform magnetic field, varying no more than 10 percent. Van Guens does not disagree with this finding. Instead, Van Guens argues that the uniform magnetic field limitation in the claim in question must be interpreted in light of the specification and the understanding of persons skilled in the NMR and MRI art. Van Guens then contends that the Japanese reference does not make the invention of the claim in question obvious because it does not teach the level of magnetic field uniformity required for NMR imaging. The court rejects this argument and states that the claim is not expressly limited to NMR or MRI apparatus. The court then holds that Van Guens cannot read an NMR limitation into the claim to justify his argument as to the meaning of the "uniform magnetic field." The *Van Guens* court, therefore, applies the rule that limitations from the specification are not read into the claims **only when** there is no dispute that the prior art discloses the claimed invention and the limitation provides further specificity to the claim in an

attempt to distinguish from the prior art. In the present application, the prior art does **not** disclose the claimed invention, and therefore the *Van Guens* rule does not apply here. However, in the interest of expediting prosecution of the case, claims 1, 8, 15, and 22 have been amended to clarify the claim language.

Second, Chang merely discloses moving code downward across branch operations within a superblock (Chang, page 268, section 2.5 "Code Scheduling Models", first paragraph), not sinking the wait instruction down the blocks globally to the end of the critical section. Even if the superblock is a critical section, moving the code downward across branch operations within a superblock does not sink the wait instruction down the blocks globally, or across the blocks, to the end of the critical section. Chang merely discloses moving an instruction downward within a superblock, not globally (or across the blocks), and not to the end of the critical section. To clarify this aspect of the invention, claims 1, 8, 15, and 22 have been amended.

Third, Shpeisman merely discloses a colored conflict graph 700 (Shpeisman, paragraph [0034], lines 1-7), not associating blocks of instructions with color information. Shpeisman explicitly discloses that the color corresponds to a hardware resource (Shpeisman, paragraph [0058], lines 1-3), and the assignment of a color to a node represents the assignment of a hardware resource to the data represented by the node (Shpeisman, paragraph [0058], lines 5-8). In contrast, the rejected claims recite "associating blocks of instructions . . . with color information," indicating that the colora correspond to the blocks of instructions, not hardware resources such as a data bank (Shpeisman, paragraph [0058], lines 1-3).

Fourth, Midkiff merely discloses a wait instruction to wait until an event occurs (Midkiff, page 1487, right column, Section IV.C "Two synchronization Instruction Sets"), not a wait instruction associated with a memory access. Midkiff explicitly discloses that the wait instruction must be used in conjunction with the set instruction which signals that the event waited for by the wait instruction has occurred (Midkiff, page 1487, right column, Section IV.C "Two synchronization Instruction Sets"). Accordingly, the wait instruction as taught by Midkiff is not associated with a memory access. To clarify this aspect of the invention, claims 1, 8, 15, and 22 have been amended.

The Examiner failed to establish a prima facie case of obviousness and failed to show there is teaching, suggestion, or motivation to combine the references. Moreover, the Examiner

failed to establish the factual inquires in the three-pronged test as required by the *Graham* factual inquires.

Therefore, Applicant believes that independent claims 1, 8, 15, and 22 and their respective dependent claims are distinguishable over the cited prior art references. Accordingly, Applicant respectfully requests the rejection under 35 U.S.C. §103(a) be withdrawn.

Conclusion

Applicant respectfully requests that a timely Notice of Allowance be issued in this case.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Dated: May 18, 2011

By / THINH V. NGUYEN /

Thinh V. Nguyen
Reg. No. 42,034
Tel.: (714) 557-3800 (Pacific Coast)

1279 Oakmead Parkway
Sunnyvale, CA 94085-4040